National Tsing Hua University
Fall 2023 11210IPT 553000
Deep Learning in Biomedical Optical Imaging
Homework 4

AUTHOR ONE[1] **張皓旻**

Student ID:112022533

Task A: Model Selection:

我選擇 DenseNet-121 和 ResNet-50

ResNet 是非常知名的模型，在架構上跟普通 CNN 很類似，但額外增加了 Residual learning 的更改，指的是每次的網路連接之間，再增加兩層前的輸出，當作本層的輸入，如圖 Fig.1，這樣模型所學習到的其實是"殘差值"，根據作者所說，這比直接學習特徵值更容易，因為不會發生梯度消失的問題，而 DenseNet 是 ResNet 的加強版，其結構如 Fig.2，除了上層的輸入外，還外加之前每一層的輸出當作本層的輸入。

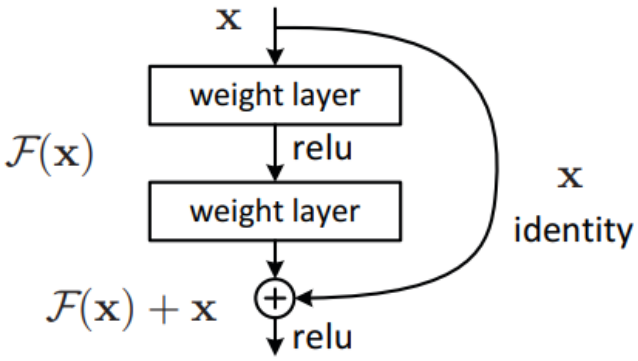

Fig.1 :ResNet 架構

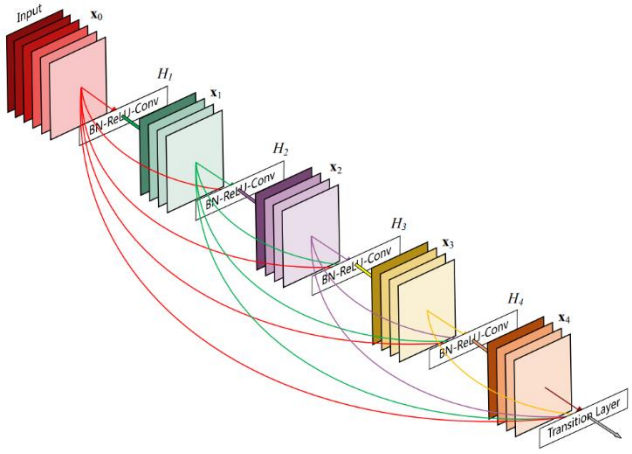

Fig.2 : DenseNet 架構

圖取自 : 1.Deep Residual Learning for Image Recognition arXiv:1512.03385v1 [cs.CV] 10 Dec 2015
2. Densely Connected Convolutional Networks arXiv:1608.06993v5 [cs.CV] 28 Jan 2018

本次選擇的是 DenseNet-121 和 ResNet-50，這兩種模型都有強大的特徵提取性能，在訓練時需要大量資料以及時間，但使用訓練好的模型，都不會太吃資源。

Task B: Fine-tuning the ConvNet:

兩種架構都可以選擇初始的 weight，在 model 設定之後加上 weight = 'IMAGENET1K_V1'之類的 weight，在 pyrtorch 上可以找到很多選擇，本次實驗 ResNet-50 使用 weight = 'IMAGENET1K_V1'， DenseNet-121 使用預設的 weight，最後需要把最後一層改成兩個輸出，就可以適用於作業要分析的胸腔 X 光影像分類(使用 CrossEntropyLoss 做二元分類)，其中要注意的是 ResNet50 最後一層叫 fc，DenseNet-121 的最後一層叫 classifier，其他的訓練參數均保持一致

(CrossEntropyLoss 、 batch_size=32 、 epochs=30 、 optimizer=Adam(lr=1e-3) 、 StepLR(step_size=10, gamma=0.1))，結果如下圖，Fig.3 是 ResNet-50 的性能表現圖，其性能非常好，最終 train acc 為 100% ，val acc 為約 98%，但 Test acc 只有約 82%，花費時間約 10 分 49 秒，Fig.4 是 DenseNet-121 的性能表現圖，其性能非常好，最最終 train acc 為 100% ，val acc 為約 98%，但之後的 Test acc 只有約 77%，花費時間約 10 分 43 秒，這兩個 model 在訓練時的表現都極佳，但 Test 時就不夠好了，可能是 Test 的數據類型比較特別，在訓練中沒看過。

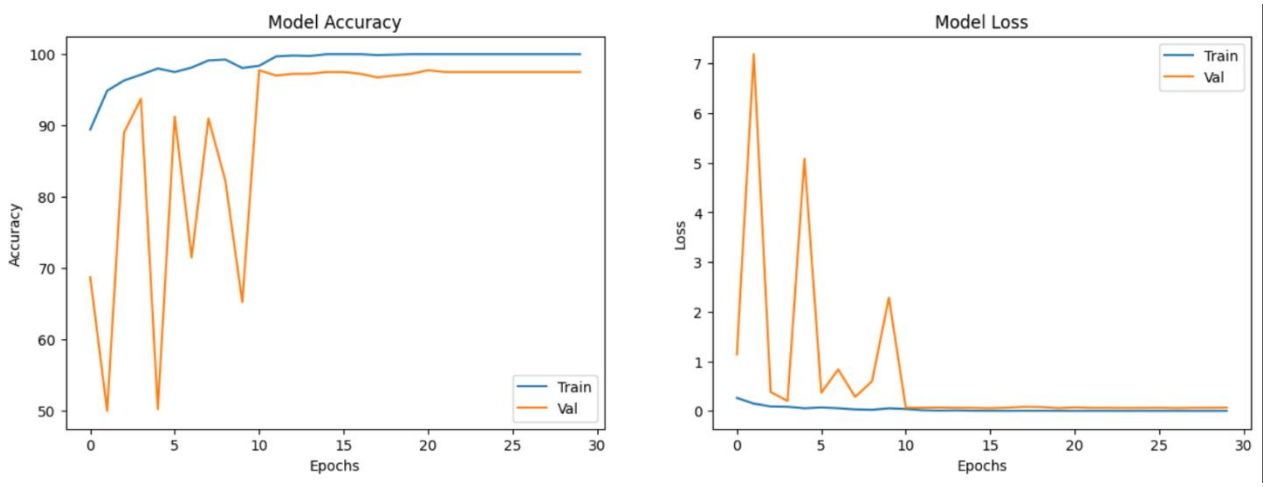

Fig.3 : ResNet 的性能表現圖

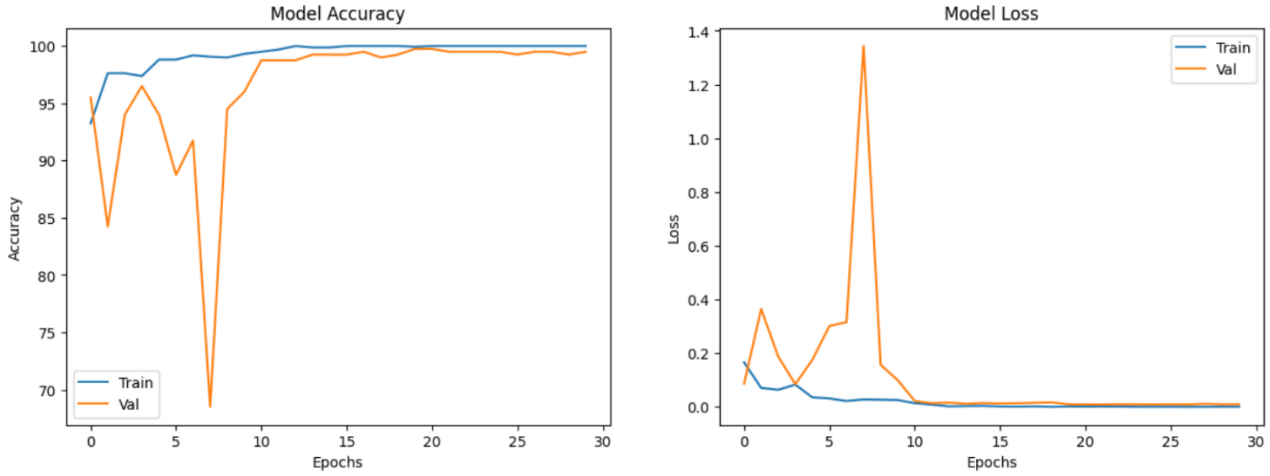

Fig.4 : DenseNet-121 的性能表現

Task C: ConvNet as Fixed Feature Extractor:

把模型中所有捲積層參數鎖定，成為特徵提取器，只有最後的全連結層能被調整，結果如下，Fig.5 是 ResNet-50 的性能圖，最終 train acc 為 96% ，val acc 為約 94%，但 Test acc 只有約 84%，花費 3 分 53 秒，Fig.6 是 DenseNet-121 的性能圖，最終 train acc 為約 96% ，val acc 為約 94%，Test acc 為 84%，花費 4 分 9 秒，可以發現訓練比起 Task B 快了非常多，這可能是因為訓練時能調的參數少了非常多，但都有 overfitting 發生。

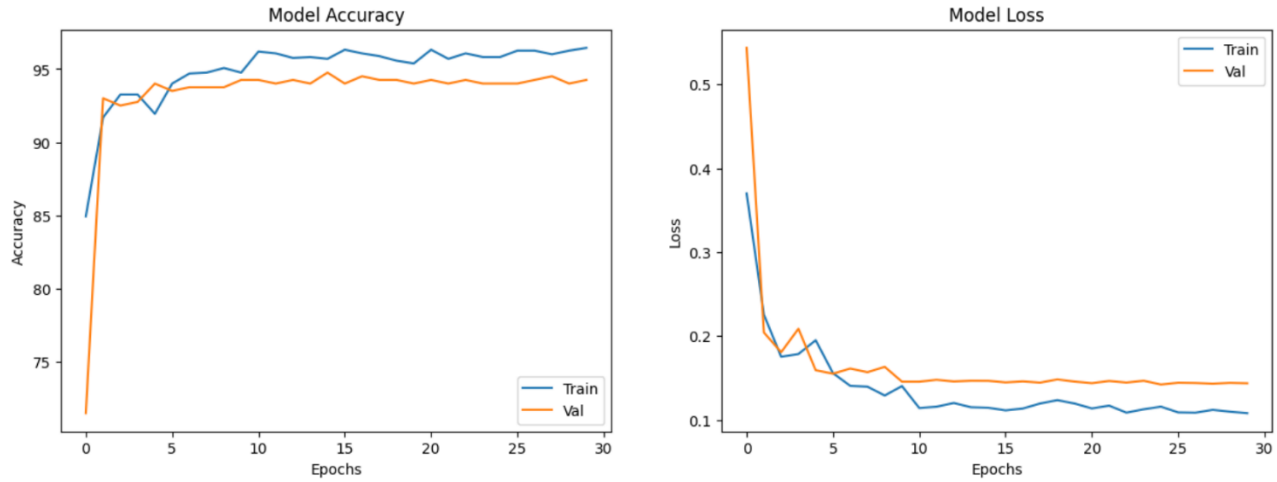

Fig.5 : ResNet-50 作為特徵提取器的性能表現

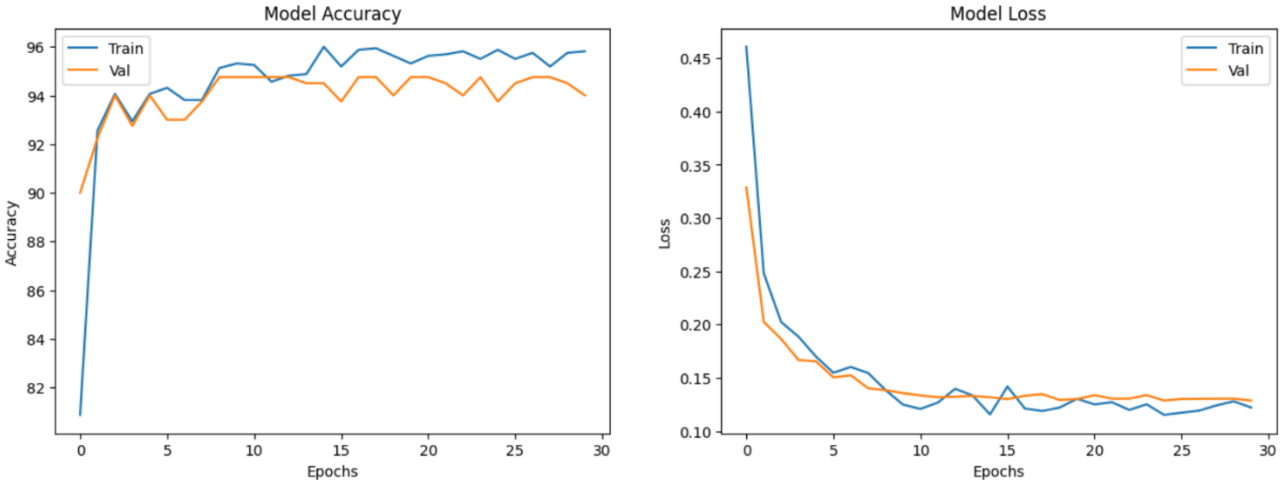

Fig.6 : DenseNet-121 作為特徵提取器的性能表現

Task D: Comparison and Analysis:

根據以上的性能表現，發現當作特徵提取器時，模型的性能可能比較弱，因為 Train acc 跟 val acc 都比可調整參數時還要低，但適應性應該更好，因為 Test acc 均高於可調整參數的模型，所以作為特徵提取器時，模型更可以提取他沒看過數據的特徵，所以有這樣的結果，而模型作為特徵提取器，還有一個優勢就是訓練速度快非常多，不用修改成千上萬的參數，讓訓練時間大幅降低。

Task E: Test Dataset Analysis:

要改善 Test acc 我想先往減少 overfitting 的方向嘗試，用 Test acc 表現比較好 DenseNet-121(特徵提取器)來做修改，首先修改 model，把最後的全連結層，多疊幾層，並加上 dropout 和 BatchNorm1d，看看能不能減少 overfitting，如下圖 Fig.7，模型性能表現如圖 Fig.8，最後的 Train acc 是 97%，val acc 是 96%，Test acc 提高到了 86%，overfitting 的情況有所改善，但 Test acc 還是沒有很好，再加上 Task B 中，DenseNet-121 的 Test acc 在訓練情況很好的同時非常的低，我推測是訓練用的數據和測試用的數據內容，不太均勻，Test 數據的長相比較特殊，在訓練時都沒看過導致。

```python
model = models.densenet121(pretrained=True)

# ConvNet as fixed feature extractor (freeze parameters)
for param in model.parameters():
        param.requires_grad = False

# num_ftrs = model.fc.in_features
num_ftrs = model.classifier.in_features

# Here the size of each output sample is set to 2.
# Alternatively, it can be generalized to ``nn.Linear(num_ftrs, len(class_names))``.

# model.classifier = nn.Linear(num_ftrs, 2)

model.classifier = nn.Sequential(nn.Linear(num_ftrs, 128),
            nn.BatchNorm1d(128),
            nn.ReLU(),
            nn.Dropout(0.5),
            nn.Linear(128, 128),
            nn.BatchNorm1d(128),
            nn.ReLU(),
            nn.Dropout(0.8),
            nn.Linear(128, 2))
```

Fig.7 :修改後的 model

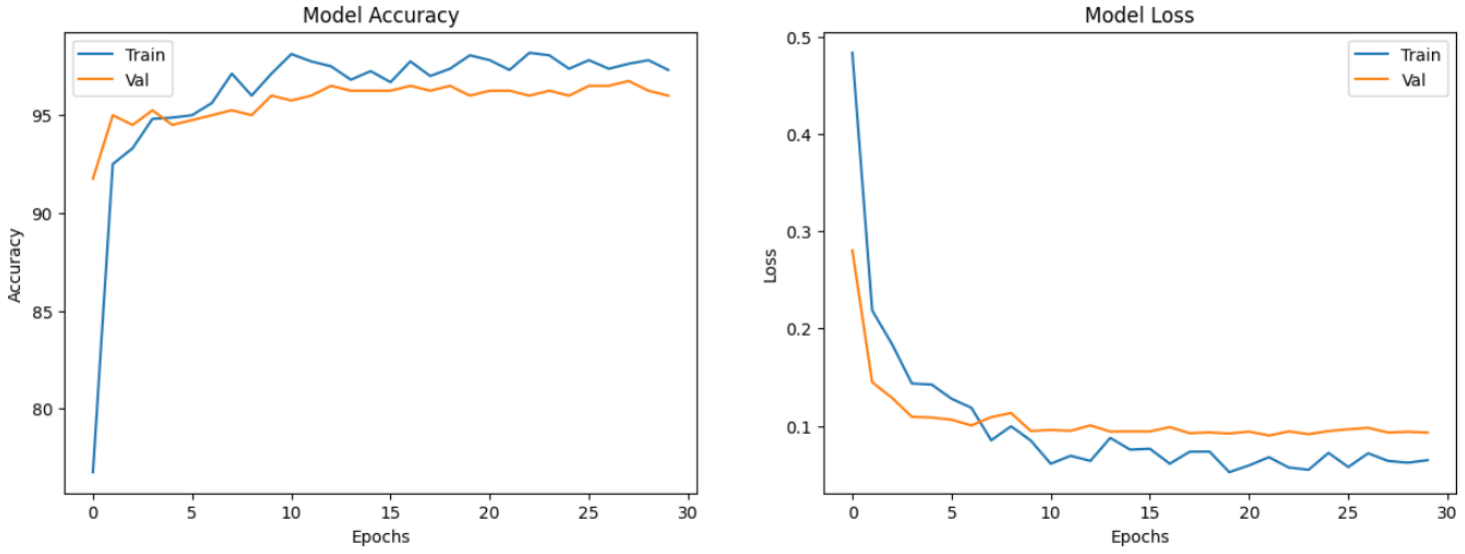

Fig.8 :修改後的 model 性能表現